

? show files;ds
File 2:INSPEC 1898-2006/Sep w1
(c) 2006 Institution of Electrical Engineers
File 5:Biosis Previews(R) 1969-2006/Sep w2
(c) 2006 The Thomson Corporation
File 6:NTIS 1964-2006/Sep w1
(c) 2006 NTIS, Intl Cpyrght All Rights Res
File 8:Ei Compendex(R) 1970-2006/Sep w1
(c) 2006 Elsevier Eng. Info. Inc.
File 15:ABI/Inform(R) 1971-2006/Sep 14
(c) 2006 ProQuest Info&Learning
File 16:Gale Group PROMT(R) 1990-2006/Sep 14
(c) 2006 The Gale Group
File 20:Dialog Global Reporter 1997-2006/Sep 15
(c) 2006 Dialog
File 24:CSA Life Sciences Abstracts 1966-2006/Aug
(c) 2006 CSA.
File 33:Aluminium Industry Abstracts 1966-2006/Aug
(c) 2006 CSA.
File 34:SciSearch(R) Cited Ref Sci 1990-2006/Sep w2
(c) 2006 The Thomson Corp
File 35:Dissertation Abs Online 1861-2006/Aug
(c) 2006 ProQuest Info&Learning
File 47:Gale Group Magazine DB(TM) 1959-2006/Sep 14
(c) 2006 The Gale group
File 62:SPIN(R) 1975-2006/Sep w1
(c) 2006 American Institute of Physics
File 63:Transport Res(TRIS) 1970-2006/Aug
(c) fmt only 2006 Dialog
File 64:Environmental Engineering Abstracts 1966-2006/Aug
(c) 2006 CSA.
File 71:ELSEVIER BIOBASE 1994-2006/Sep w2
(c) 2006 Elsevier B.V.
File 73:EMBASE 1974-2006/Sep 15
(c) 2006 Elsevier B.V.
File 88:Gale Group Business A.R.T.S. 1976-2006/Sep 14
(c) 2006 The Gale Group
File 103:Energy SciTec 1974-2006/Jul B2
(c) 2006 Contains copyrighted material
File 104:AeroBase 1999-2006/July
(c) 2006 Contains copyrighted material
File 141:Readers Guide 1983-2006/Jul
(c) 2006 The HW Wilson Co
File 144:Pascal 1973-2006/Aug w3
(c) 2006 INIST/CNRS
File 148:Gale Group Trade & Industry DB 1976-2006/Sep 15
(c) 2006 The Gale Group
File 155:MEDLINE(R) 1950-2006/Sep 14
(c) format only 2006 Dialog
File 159:Cancerlit 1975-2002/Oct
(c) format only 2002 Dialog
File 213:ONTAP(R) INSPEC
(c) 1989 Institution of Electrical Engineers
File 239:Mathsci 1940-2006/Oct
(c) 2006 American Mathematical Society
File 275:Gale Group Computer DB(TM) 1983-2006/Sep 14
(c) 2006 The Gale Group
File 292:GEOBASE(TM) 1980-2006/Sep w1
(c) 2006 Elsevier B.V.
File 323:RAPRA Rubber & Plastics 1972-2006/Aug
(c) 2006 RAPRA Technology Ltd
File 324:German Patents Fulltext 1967-200636
(c) 2006 Univentio
File 348:EUROPEAN PATENTS 1978-2006/ 200637
(c) 2006 European Patent Office

File 399:CA SEARCH(R) 1967-2006/UD=14512
(c) 2006 American Chemical Society
File 427:Fort Worth Star-Telegram 1993-2004/Feb 25
(c) 2004 Fort Worth Papers
File 434:SciSearch(R) Cited Ref Sci 1974-1989/Dec
(c) 2006 The Thomson Corp
File 435:Art Abstracts 1984-2006/Aug
(c) 2006 The HW Wilson Co
File 440:Current Contents Search(R) 1990-2006/Sep 15
(c) 2006 The Thomson Corp
File 471:New York Times Fulltext 1980-2006/Sep 15
(c) 2006 The New York Times
File 474:New York Times Abs 1969-2006/Sep 14
(c) 2006 The New York Times
File 484:Periodical Abs Plustext 1986-2006/Sep w2
(c) 2006 ProQuest
File 492:Arizona Repub/Phoenix Gaz 19862002/Jan 06
(c) 2002 Phoenix Newspapers
File 570:Gale Group MARS(R) 1984-2006/Sep 14
(c) 2006 The Gale Group
File 582:Augusta Chronicle 1996- 2006/Sep 14
(c) 2006 Augusta Chronicle
File 608:KR/T Bus.News. 1992-2006/Sep 15
(c)2006 Knight Ridder/Tribune Bus News
File 619:Asia Intelligence Wire 1995-2006/Sep 13
(c) 2006 Fin. Times Ltd
File 621:Gale Group New Prod.Annou.(R) 1985-2006/Sep 14
(c) 2006 The Gale Group
File 633:Phil.Inquirer 1983-2006/Sep 14
(c) 2006 Philadelphia Newspapers Inc
File 634:San Jose Mercury Jun 1985-2006/Sep 14
(c) 2006 San Jose Mercury News
File 647:CMP Computer Fulltext 1988-2006/Oct w5
(c) 2006 CMP Media, LLC
File 654:US Pat.Full. 1976-2006/Sep 12
(c) Format only 2006 Dialog
File 706:(New Orleans)Times Picayune 1989-2006/Sep 14
(c) 2006 Times Picayune
File 710:Times/Sun.Times(London) Jun 1988-2006/Sep 15
(c) 2006 Times Newspapers
File 711:Independent(London) Sep 1988-2006/Sep 14
(c) 2006 Newspaper Publ. PLC
File 713:Atlanta J/Const. 1989-2006/Sep 14
(c) 2006 Atlanta Newspapers
File 717:The Washington Times Jun 1989-2006/Sep 14
(c) 2006 Washington Times
File 727:Canadian Newspapers 1990-2006/Sep 15
(c) 2006 Southam Inc.
File 728:Asia/Pac News 1994-2005/Dec w2
(c) 2005 Dialog
File 732:San Francisco Exam. 1990- 2000/Nov 21
(c) 2000 San Francisco Examiner
File 740:(Memphis)Comm.Appeal 1990-2006/Sep 13
(c) 2006 The Commercial Appeal
File 743:(New Jersey)The Record 1989-2006/Sep 14
(c) 2006 No.Jersey Media G Inc

Set	Items	Description
S1	173	(REVERS? OR INVERT? OR INSIDE()OUT OR MIRROR? OR BACKWARD?- (3N)(STAR OR STARS OR SNOWFLAKE? ? OR SNOW()FLAKE? ?)(10N)(D- TD OR DEFINITION OR DICTIONARY OR MODEL?) NOT PY>1999
S2	127	RD (unique items)

? t2/3,k/all
>>>KWIC option is not available in file(s): 399

? show files;ds

File 2:INSPEC 1898-2006/Sep w1
(c) 2006 Institution of Electrical Engineers
File 8:EI Compendex(R) 1970-2006/Sep w1
(c) 2006 Elsevier Eng. Info. Inc.
File 9:Business & Industry(R) Jul/1994-2006/Sep 14
(c) 2006 The Gale Group
File 13:BAMP 2006/Sep w1
(c) 2006 The Gale Group
File 15:ABI/Inform(R) 1971-2006/Sep 15
(c) 2006 ProQuest Info&Learning
File 16:Gale Group PROMT(R) 1990-2006/Sep 14
(c) 2006 The Gale Group
File 75:TGG Management Contents(R) 86-2006/Sep w1
(c) 2006 The Gale Group
File 88:Gale Group Business A.R.T.S. 1976-2006/Sep 14
(c) 2006 The Gale Group
File 144:Pascal 1973-2006/Aug w3
(c) 2006 INIST/CNRS
File 148:Gale Group Trade & Industry DB 1976-2006/Sep 15
(c) 2006 The Gale Group
File 275:Gale Group Computer DB(TM) 1983-2006/Sep 14
(c) 2006 The Gale Group
File 280:ONTAP Derwent World Patents Index
(c) 2006 The Thomson Corp.
File 340:CLAIMS(R)/US Patent 1950-06/Sep 14
(c) 2006 IFI/CLAIMS(R)
File 348:EUROPEAN PATENTS 1978-2006/ 200637
(c) 2006 European Patent Office
File 349:PCT FULLTEXT 1979-2006/UB=20060914UT=20060907
(c) 2006 WIPO/Thomson
File 351:Derwent WPI 1963-2006/UD=200658
(c) 2006 The Thomson Corporation
File 440:Current Contents Search(R) 1990-2006/Sep 15
(c) 2006 The Thomson Corp
File 484:Periodical Abs Plustext 1986-2006/Sep w2
(c) 2006 ProQuest
File 570:Gale Group MARS(R) 1984-2006/Sep 14
(c) 2006 The Gale Group
File 619:Asia Intelligence Wire 1995-2006/Sep 13
(c) 2006 Fin. Times Ltd
File 647:CMP Computer Fulltext 1988-2006/Oct w5
(c) 2006 CMP Media, LLC
File 649:Gale Group Newswire ASAP(TM) 2006/Sep 01
(c) 2006 The Gale Group
File 654:US Pat.Full. 1976-2006/Sep 12
(c) Format only 2006 Dialog
File 767:Frost & Sullivan Market Eng 2006/Aug
(c) 2006 Frost & Sullivan Inc.
File 990:NewsRoom Current Mar 1 -2006/Sep 15
(c) 2006 Dialog
File 994:NewsRoom 2003
(c) 2006 Dialog
File 996:NewsRoom 2000-2001
(c) 2006 Dialog

Set	Items	Description
S1	172	DIMENSION()TABLE(30N)SCHEMA(30N)((FACT OR TRUTH)()TABLE? ?)
S2	41	S1 NOT PY>1999
S3	34	RD (unique items)

? t3/3,k/all

3/3,k/1 (Item 1 from file: 16)
DIALOG(R)File 16:Gale Group PROMT(R)
(c) 2006 The Gale Group. All rts. reserv.

? t 05949017/7

05949017/7

DIALOG(R)File 16:Gale Group PROMT(R)
(c) 2006 The Gale Group. All rts. reserv.

05949017 Supplier Number: 53209263 (THIS IS THE FULLTEXT)
The Schema Wars.(Technology Information)

Craig, Robert
ENT, p30(1)
Nov 18, 1998

TEXT:

There's a debate going on in the decision support world on the topic of the most appropriate data model to use for a relational decision support database. Two camps are promoting their respective points of view: the third normal form (3NF) camp and the star schema camp.

First, a little bit of background. Relational databases, which are the industry norm for storing data from transaction processing applications, are based on a mathematical model that describes tables, columns, and their relationships. Normalization is the process of defining the relationships between tables and columns. There are several levels of normalization, but it is generally accepted that third normal form is the structure that is most appropriate for transaction-processing databases.

In a 3NF model, every nonkey column is dependent on one, and only one, key column; it is dependent on the entire key, not a subset of a key; and there are no repeating groups of columns within a table. Nonkey columns within a table can be, and often are, keys in other tables. These are called foreign keys. An order table will have order-related information with "order_number" as the primary key. A "customer_id" column is a foreign key that points to a primary key in the customer table.

However, as they say, the devil is in the details. It turns out that when most modelers implement their logical normalized model as a physical model, the result is to create hundreds, or even thousands, of tables. It quickly becomes apparent to many organizations that end users can be completely baffled when they are presented with unfettered access to a fully normalized database. They have no idea where to start or how to search, browse and navigate the database to locate the data that is meaningful to them.

In addition, a typical decision support query can stress the system in a variety of ways. Decision support queries often need to look at data from multiple tables, which forces a multitable join. Users want to aggregate data, which requires the database to perform sums, counts and other calculations. Often the database engine must scan very large amounts of data. Finally, the data needs to be sorted before it's presented to the user. These activities -- joins, aggregation, scanning, and sorting -- are all very resource intensive. Some databases perform them better than others.

The star schema, which provides a multidimensional view of the data based on familiar relational technology, was designed to alleviate these complexities. A star schema has a central table -- the fact table -- and dimensional tables. Each dimension table has a single-part primary key. The fact table has a multipart primary key, which is a concatenation of the dimension table keys. The fact table contains atomic, detail-level facts, usually numerical measures, while the dimension tables contain descriptive data.

Since the star schema provides a hierarchical, multidimensional view of the data, it is easier for end users to navigate. In addition, the dimensional model takes care of aggregates and reduces the number of joins. But like OLAP engines, the star schema requires the developer to explicitly define and build the dimensions, which means that he or she has to have a good idea about what questions the users will want to ask. One of the strengths of the 3NF model is that any query can be launched against the database, as long as the data is there. Therefore, the star schema is good for routine categories of questions, while 3NF makes more sense for ad hoc, unanticipated questions.

Ginger R. DeMille

Given this disparity of presentation, which approach should a database designer adopt? I think the thing to do is support both. Your power users and more sophisticated business analysts need the ability to ask any question of the database. These people, who are usually highly trained and experienced, are not likely to be hindered by the number of tables they need to understand. On the other hand, you need to provide an easy-to-navigate interface that enables less-sophisticated users to easily browse the database. For these users you can, and should, create star schema views. This approach lets you create different views for different categories of end users, based on their analytical needs, without limiting overall database flexibility for power users. --Robert Craig is director of the Data Warehousing and Business Intelligence Division at Hurwitz Group Inc. (Framingham, Mass.). Contact him at rcraig@hurwitz.com or via the web at www.hurwitz.com.

COPYRIGHT 1998 Boucher Communications, Inc.

COPYRIGHT 1999 Gale Group

?

[Back](#)**5 page(s) will be printed.**

Record: 1

Title: Designing a data mart.
Authors: Gagnon, Gabrielle
Source: PC Magazine; 11/16/99, Vol. 18 Issue 20, p213, 4p, 3 diagrams
Document Type: Article
Subject Terms: *DATA warehousing
DESIGN
DATA marts
Abstract: Shows the basic steps in designing a data mart for data warehousing. Advantages of data marts which make them popular; Difference of data marts from larger warehouses; data marts as Online Analytical Processing tools; Dimensional model of data marts.
Full Text Word Count: 2181
ISSN: 0888-8507
Accession Number: 2474671
Database: Business Source Corporate

Section: Departmental Solutions
DESIGNING A DATA MART

With their tighter focus, data marts are easier to design than data warehouses. We show you the basics.

Data marts are by far the most popular and successful data warehouses today. Their tight focus, incremental development approach, and high return on investment make them appealing to competitive business users, harried IT professionals, and cost-conscious managers alike.

Like most data warehouses, data marts are decision-support systems. They differ from larger warehouses in that they are limited in scope, focusing on a single department or business process. This focus enables data marts to be deployed much faster than full-blown enterprise data warehouses, which often take years of analysis and planning.

With consistent formatting, data marts can even be linked together to form a distributed enterprise warehouse. (This is sometimes referred to as the bottom-up approach to warehouse design.) In fact, many designers recommend taking such an incremental approach, because although the whole project may take several years, the individual data marts begin performing and adding value right away.

But whether you want to set up a departmental stovepipe that will never connect to anything else or integrate a distributed warehouse across a global enterprise, data marts have fundamental design requirements that differ from operations systems. Let's take a look.

OLTP VS. OLAP

Data marts are not like Online Transactional Processing (OLTP) systems that handle the day-to-day operations of a business. They are decision-support systems that help business analysts identify trends, make comparisons and predict future results. These functions are often referred to as Online Analytical Processing (OLAP).

OLAP functions can't readily be performed on OLTP systems. For one thing, OLTP systems keep only a

snapshot of the most current information online; OLAP requires a history, of transactions over long periods of time. OLTP systems are constantly being updated and sometimes have missing or erroneous data; OLAP requires static, complete data, adjusted for errors. Finally, OLTP systems process thousands (sometimes millions) of transactions per day; OLAP systems process only one transaction per download cycle--when data is batch-loaded into the system--but may process thousands of concurrent queries per day. (Some people regard queries as transactions, but we don't. A transaction takes a database from one consistent state to another, which implies change.)

These differences in function translate to differences in design. OLTP systems typically use the entity-relationship model familiar to relational database administrators. The entity-relationship model normalizes data to minimize redundancy and is optimal for data entry and updates. But OLAP systems are rarely updated by users and are used primarily for queries and reports. While it is possible to design data marts based on the entity-relationship model, many experts recommend using a dimensional model instead. The dimensional model organizes data for easy retrieval and is optimal for reporting.

THE DIMENSIONAL MODEL

The dimensional model takes a business process like marketing or sales and organizes it by dimensions, such as products or sales regions. The dimensional model is represented as a hypercube or multidimensional array (see Figure 1). This shows how users can slice and dice data any way they choose.

Values in the array are called facts, and they are used to measure performance. Descriptions of the facts are called dimensions, and they function as constraints or as row headings.

In a sales analysis data mart, facts might be numbers of units sold or total sales, and dimensions might be product names or store locations. Because business analysts want to look at data historically, time is typically also a dimension.

Each block in the hypercube represents a value. Values may be found via any dimension or combination of dimensions. In our sales analysis example, if you wanted to find out how many widgets were sold during the past year, you could sum widget sales along the time dimension. If you wanted to know which sales region performed best overall, you could search regions and find the maximum total sales.

Several OLAP tools work with nonrelational dimensional databases, but many also work with relational databases designed on dimensional principles. OLAP products built for such databases are called Relational OLAP (ROLAP) tools.

STAR SCHEMA

A dimensional model implemented in a relational system is called a star join schema, or star schema for short (See Figure 2). In a star schema, one central fact table contains all of the attributes to be measured. The rest of the tables are dimension tables, which contain descriptive attributes. The fact table key (or unique identifier) is a composite of all the dimension table keys. This enables the fact table to join to all of the dimension tables. The dimension tables, however, are capable of joining only to the fact table. (Relationship diagrams depict the fact table as being the hub of concentric dimension tables arranged in the pattern of a star, hence the name.)

The simplicity of the star schema makes retrieval very efficient, and this appeals to end users who don't like to navigate through a maze of relationships. Let's take a look at the steps involved in designing a star schema data mart.

BUILDING A STAR SCHEMA DATA MART

Determine the business process to analyze. The first step in designing a data mart is to decide what business process to model. A business process is a group of related activities that support a business function, such as marketing or inventory. Consider the types of questions users will want the data mart to answer. A sales team may want to measure their performance against that of competitors. A marketing department may want to find out the effectiveness of various marketing campaigns.

Determine facts and dimensions. Once you know what questions the data mart should answer, you can select the facts and dimensions best suited to answer them. What sort of facts work best, and how can you tell a fact from a dimension?

Facts are items to be measured or analyzed. Ralph Kimball, the foremost authority on dimensional modeling, says the best facts are "numeric, continuously valued, and additive."

Numeric facts represent quantities. (They don't include items like social security numbers or zip codes, which can be classified as numeric data types to enforce integrity rules.) Numeric facts can have various mathematical operations performed on them; they are easy to measure.

A continuously valued fact can be any one of a range of values, and it usually changes frequently. If you counted the number of ice cream cones a vendor sold in one hour on a hot summer day and then counted again an hour later, chances are the number would be different. You wouldn't know the value unless you took the measurement, thus the number of ice cream cones sold would be an example of a continuously valued fact. Continuously valued facts are useful because they always tell you something new. (There's not much point in tracking something that doesn't change.)

An additive fact is capable of being added in all the dimensions. For example, in our sales analysis data mart in Figure 1, the number of ice cream cones sold would be an additive fact, because you can add up how many ice cream cones were sold last week (by time): how many were sold in California (by region): and how many were chocolate chip (by product). Kimball favors additive facts because they produce compact result sets. Queries against a historical database can easily scan thousands of records, and the ability to summarize their contents by adding them together is a valuable asset.

Items that are numeric, continuously valued, and additive are almost always included in facts tables, but not all facts have to fit that profile.

Some facts are semiadditive. That is, they may be logically added along some dimensions but not others. For example, it makes sense to add inventory levels by product or by region, but not over time. (The number of vanilla ice cream cartons on hand today plus the number on hand last week is a meaningless calculation.) Although semiadditive facts can't be added across the board, they may be measured in other ways. Averaging inventory levels along the time dimension does make sense, as does finding minimum and maximum levels and some other statistical functions.

Finally, there are nonadditive facts. These facts can't be logically added in any dimension. A ratio is an example of a numeric, nonadditive fact. Nonnumeric facts are also nonadditive. Nonadditive facts can't be added, but they can be counted, so they are still measurable. Nonadditive items are more often grouped with dimensions, though.

As we mentioned, dimensions are descriptive attributes, serving as constraints in queries or as row headings in reports. In a star schema, dimensions are also the tables that contain such attributes. Dimensions are usually textual, relatively static, and nonadditive. A person's name and social security number are examples of dimensions. A person's name may change, but it is not likely to change often, and it's not something you'd ordinarily measure. Rather, it tells you about something you might be measuring in the fact table, like employee pay raises.

Distinguishing facts from dimensions is not always easy. Sometimes you might include numeric, additive items, such as unit cost, in a dimension table. And sometimes static or nonadditive values are appropriate in the fact table. It all depends on the application. For example, gender would usually be considered a dimension--it is textual, static, and nonadditive--but if you were designing a human resource data mart and analysts frequently asked questions like "What is the ratio of male to female employees in managerial positions?", you might put gender in the fact table. Even the experts disagree on what goes where sometimes.

Determine the grain. The next step is to determine the grain. The grain is the lowest level of detail stored

In the data mart.

Information can be summarized on different levels according to hierarchical groupings (see Figure 3). Daily sales can be summarized by weekly, monthly, or yearly totals; items sold may be summarized by product, product category, or product line. Sometimes more than one hierarchy can apply. Stores may be grouped by postal regions for shipping rates or by sales territories for marketing campaigns. You can store both hierarchies in the same dimension.

Data marts let users work with different levels of summary data. The question is, how much detail do users need?

Because there's always the chance someone will want to drill down to a base transaction, you might think you should always include the lowest level of detail, but that's usually not necessary and often impractical.

If a company had 1 million customers, and each customer produced ten transactions a month, after five years that would equal 600 million transactions. Assuming each transaction used 200 bytes, transactions alone would need 1.2 terabytes of storage. For trend analysis, you might keep ten or more years online, which would require a minimum of 2.4 terabytes. Thus, even a small transaction file can become very large when viewed historically. And when you add aggregated summary data and indexes, the data mart could easily double or triple in size.

Before buying petabytes of disk space, consider the business questions you're trying to answer, and choose the grain accordingly. If you operate a retail outlet and want to track inventory movement, you don't have to keep every customer receipt in the data mart; you can summarize purchases on the product level. If, you want to analyze customer buying habits, however, you'll need the individual receipts (as well as a way of linking specific customer profiles to the records).

Sometimes the rate of change gives you a clue as to what grain to use: If you were tracking employee pay history, it wouldn't make much sense to use a daily or weekly grain when employees get annual performance reviews.

Organize dimensions into tables. Once you've determined the grain, the question of what other attributes belong in the dimension tables can be answered. Time is always a dimension in a historic data mart, and you should put the time units that apply to your application in the Time dimension table. Days, weeks, months, and years are obvious choices if the grain is on a daily transaction level. Months and years might be all you need if the application grain calls for month-end and year-end summaries. Quarters could be included in financial data marts, and holiday flags might be useful in human resource systems.

In this article, we've offered a fairly general overview of data marts and their uses. We've only touched on some issues like handling changes over time, and we haven't even mentioned how to select keys. But we hope we've given you an idea of what's involved in designing a data mart and have piqued your curiosity to explore the subject further.

~~~~~

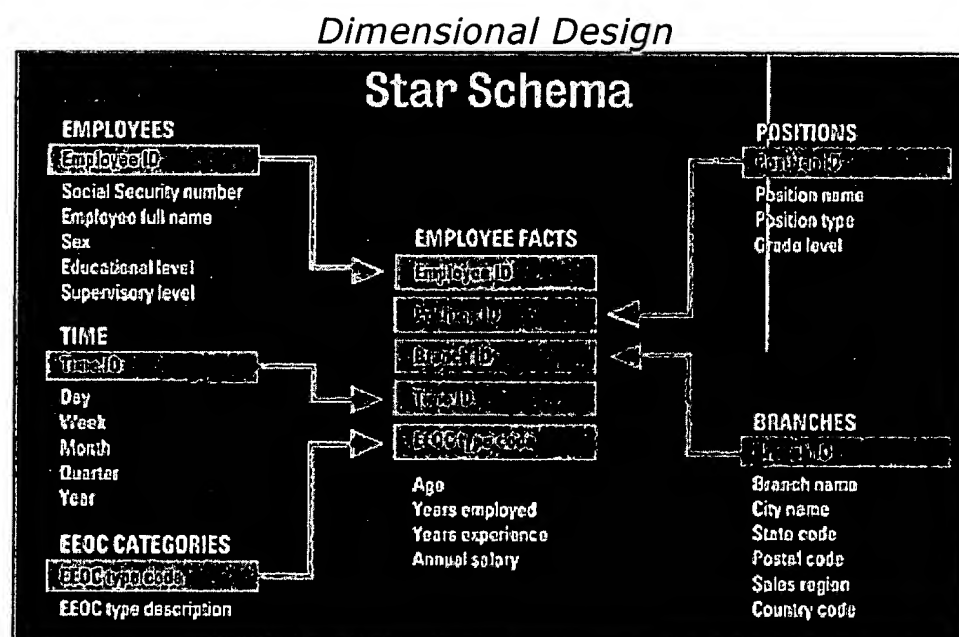
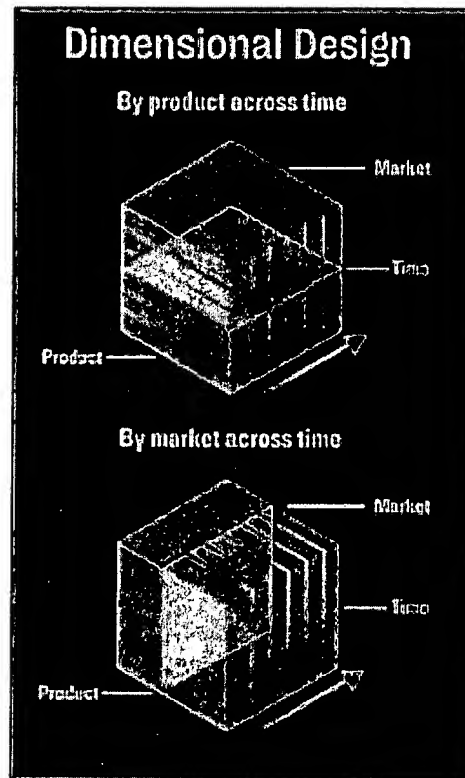
By Gabrielle Gagnon ributing Editor of PC Magazine.; Sally Wiener Grotta, Contributing Editor of PC Magazine.; M. David Stone, Contributing Editor of PC Magazine.; Bruce Brown, Contributing Editor of PC Magazine.; Jennifer Triverio, Associate Editor and S. Jae Yang, PC Magazine Labs Project Leader

Gabrielle Gagnon is a frequent contributor to PC Magazine.

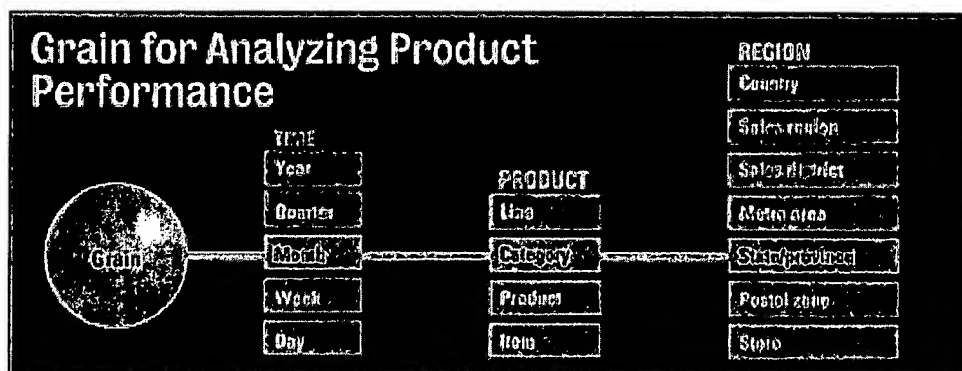
---

Copyright of PC Magazine is the property of ZDNet and its content may not be copied or emailed to multiple

sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.



*Star Schema*



*Grain for Analyzing Product Performance*

[Back](#)